

Системно програмиране

Курсова работа

Изработил: Георги Николов

Фак. номер: 121216066

група: 48

Анализ на изготвеното приложение

Изготвено е приложение(конзолно такова). За работата му е дадено следното условие:

Да се организира къпане на студенти след студентски купон, като трябва да бъдат спазени следните правила:

- Студентите да бъдат изкъпани на порции по N човека(капацитета на банята).
- Да се осигури пребиваване в банята **само на студенти от един и същи пол.**
- Да се осигури синхронизация на къпането, а именно студенти от кой пол първи ще влезнат в банята зависи от това дали представители му са повече от представителите на другия пол.

Начин на работа на изготвеното приложение:

чакащите за къпане студенти са заредени от текстов файл в свързан списък. **Определя се броя на хората от двата пола**(по този начин се съблюдава гореописаното условие в банята да влизат първи представителите на пола с повече пердставители). Определя се критична секция(самата баня), която може да бъде достъпена само от представители на единия пол (само от една нишка). За синхронизацията между различните групи полове се грижат две нишки: такава "къпеща мъжете" и такава "къпеща жените". Всяка от двете нишки има опция или веднага да започне къпането на хората от дадения пол, или да започне да слуша кога другата нишка ще освободи банята.

Комуникацията м/у нишките става чрез `message_queue`, предава се съобщението : "SEND!" , когато банята е освободена и може да се достъпи от другата нишка. Самият процес на къпане се осъществява, като група от K човека се "подава" на банята. Възможни са следните

варианти:

- $K > N$ в такъв случай къпането става на подгрупи, докато не бъдат изкъпани всички студенти от дадения пол.
- $K \leq N$ в такъв случай къпането на студентите от даден пол става наведнъж.
- $K == 0$ не се изпълнява къпане, изкарва се съобщение, описващо че няма представители от дадения пол за къпане.

По време на работа на програмата, след всяко изкъпване на студент, същият бива записван в отделен файл. По този начин се осигурява съхранение на вече изкъпаните студенти и се избягва евентуалното им повторно изкъпване при пускане програмата отново.

Функционално описание на приложението

Първата част се състои в извличане на

студентите от файла - "students.txt" за целта са използвани системните функции open/read. Поради използването на функцията read се налага използването на буфер(buffer), в който се записва съдържанието на файла. Следваща стъпка е разпределянето на съдържанието от файла на студентите. Създадена е структурата Student, т.е. съставката на свързания списък от студенти. Чрез използването на функцията strtok() се разделя съдържанието от файла. Заделя се динамично памет за двумерния масив: char (* students)[148] , неговата цел е да съхрани данните за всички студенти. Следва повторно използване на strtok(), този път, за да се раздели името от пола на студента. При всяко "разделяне" се създава и попълва нов елемент от свързания списък(функцията void loadListData()).

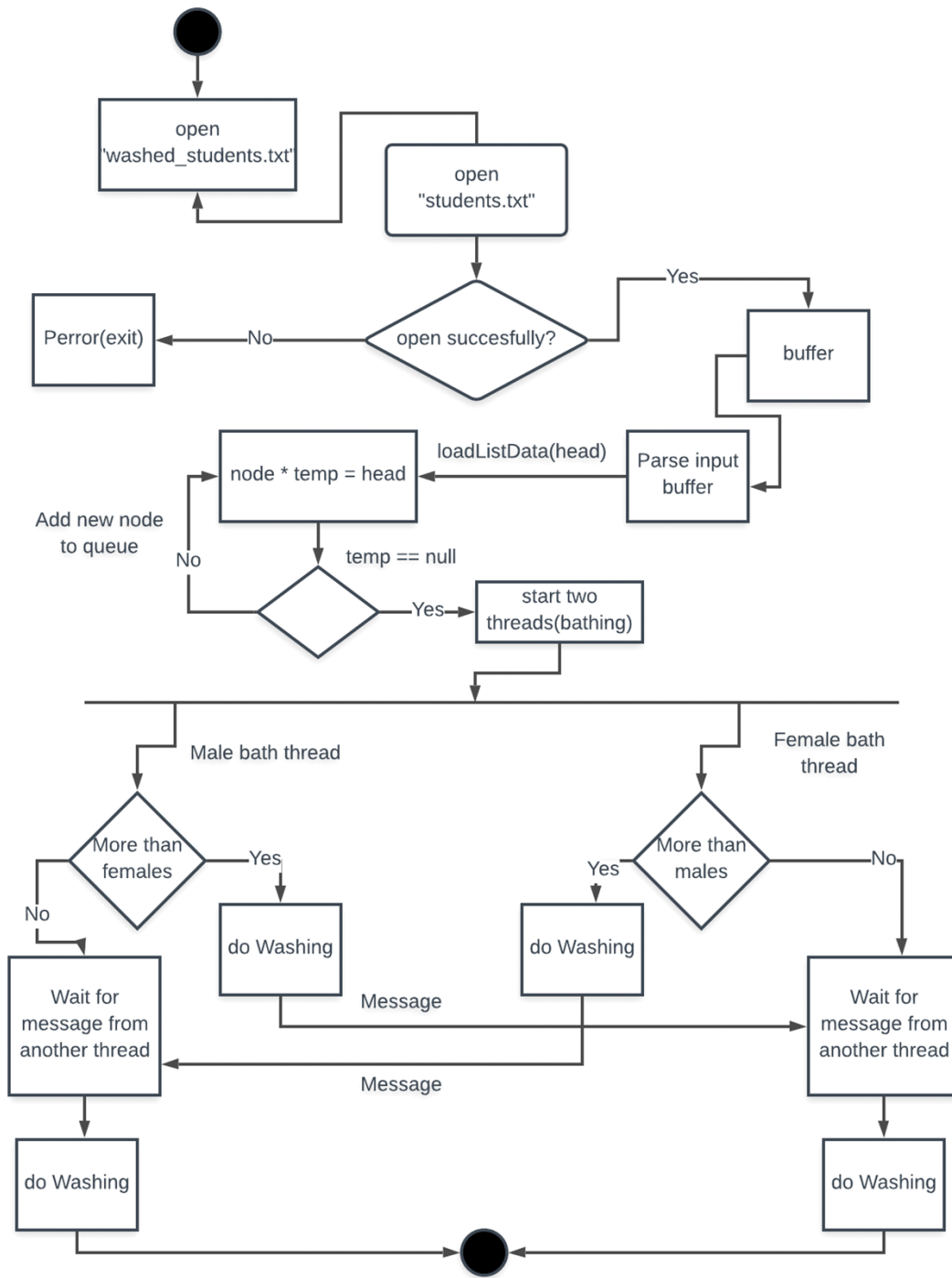
Втората част се състои в разделянето на работата между двете нишки(void* washMales(void *arg) и

`void* washFemales(void *arg))`. Двете нишки се създават и `"join"` - ват една след друга. Преди тяхното създаване бива създадена и отворена `message` - опашка, служеща като канал за комуникация м/у тях. За нейното създаване се използват флаговете: `O_RDWR`, `O_NONBLOCK` и `O_CREAT`. Тоест искаме създадената опашка да работи в режим `nonblocking`, както и да може да се чете и пише в нея. Следва започване работата на двете нишки, всяка една от тях изважда броя на конкретния пол от списъка със студенти. На база на това дали е по-голям от срещуположния или не, се стартира директно къпането, или се започва чакане на съобщение от другата нишка(безкраен цикъл). Когато съобщението, че банята е свободна бъде получено, чакането приключва и започва процеса на къпане. С други думи всяка от нишките или директно извиква функцията: `void startWashing(node * head, Gender gender)`; или започва да "цикли" в безкраен цикъл, докато не получи съобщение, след

получаването на което се вика функцията: `void startWashing(node * head, Gender gender);`

Трета част - обяснява самия процес на къпане. Както горе бе описано процесът на къпане се изпълнява в зависимост колко подгрупи има дадена група от мъже или жени. За да могат да бъдат отсети вече изкъпаните студенти във функцията `loadListData()` се прави проверка дали името на даден студент фигурира във файла с изкъпани студенти, ако да, то дадения не се добавя като елемент на списъка.

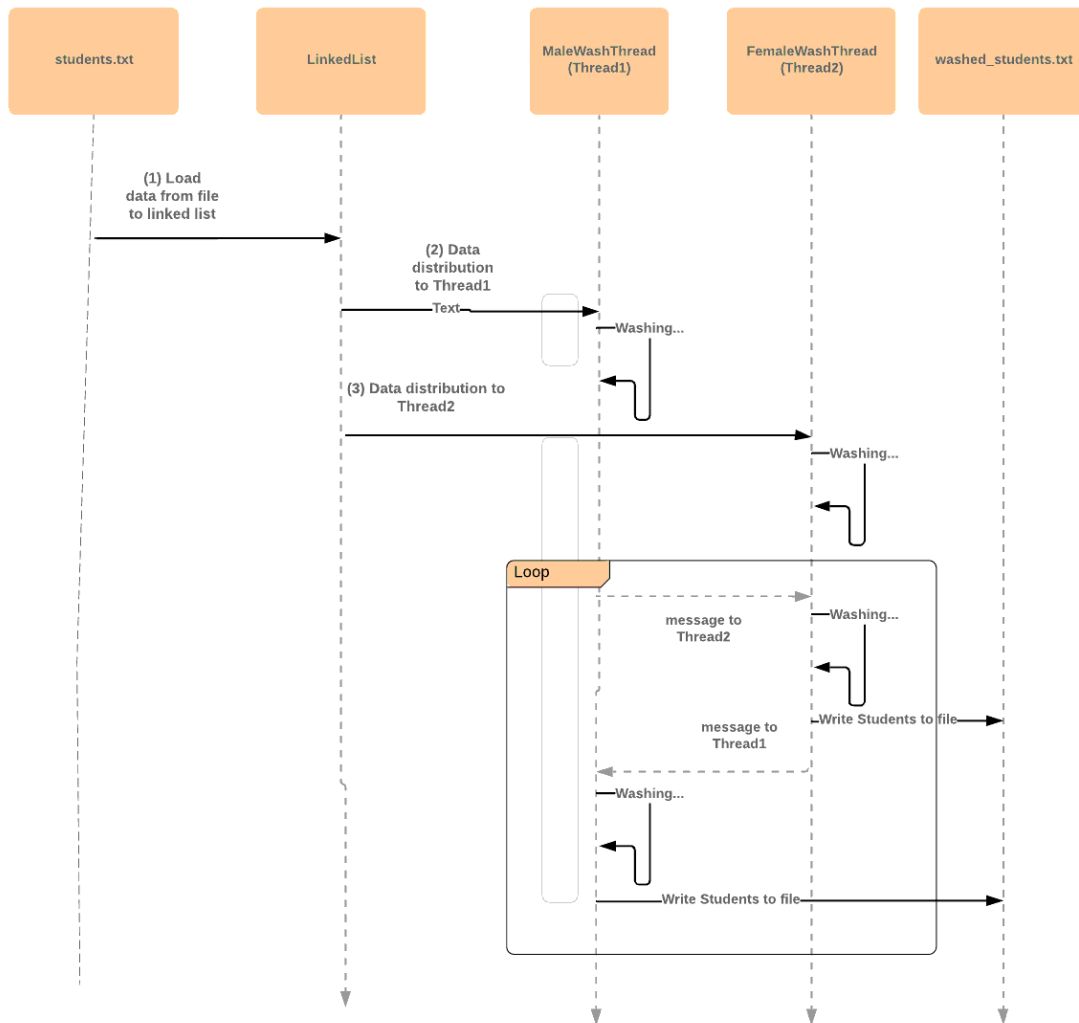
UML - activity diagram



UML - sequence diagram

SP

G. N. | May 7, 2019



Изпълнение на функционалностите

- **void* washMales(void *arg):** функцията създава нишка, обработваща мъжете от опашката. Функцията може да приема като параметър и да връща всякакъв тип. В зависимост от броя на мъжете функцията започва тяхното къпане или започва да слуша, в безкраен цикъл, за message от другата нишка(void* washFemales).
- **void* washFemales(void *arg):** същата функционалност като горната функция, но за жените.
- **void startWashing(node * head, Gender gender):** функцията цели изкъпването на дадена група от даден пол. Приема за параметри: главния възел на опашката(head) и дефиниран, изброим тип "Gender". Функцията итерира през опашката, като след като бъде намерен броя хора от подадения като параметър **gender** функцията се заема с тяхното къпане. Събира се максималният

брой хора, можещи да влязат в банята(с разм. N), след което започва тяхното къпане: извикващата нишка се приспива за 1 секунда. Даденото се изпълнява, докато не приключи изпъпването на всички представители на дадения пол.

- **bool canStartWashing(node * head, Gender gender):** функция от булев тип, приема главния възел на опашката и тип Gender. Връща true, ако представителите на подадения като параметър пол са повече от представителите на срещуположния.
- **void addNode(node * head, char * name, Gender gender):** функцията добавя елемент(студент) към опашката. Създава се нов възел - **temp** на него се прикачат данните name и gender . Итериращ се през целия списък и указателя на последния елемент се слага да сочи към temp.
- **void readData(char * buffer, int fd):**

функцията приема като вход указател към тип `char` и файлов дескриптор тип `int`. Целта е информацията от `fd` да се чете, докато функцията `read()` не върне 0 байта. Всеки прочетен символ се записва в `buffer`. Тук е важно да се отбележи динамичното разширяване на `buffer`, всеки път с по 1 символ.

- **`int parseInputData()`**: целта на функцията е да отвори файла "`students.txt`", да копира съдържанието му в буфера `buffer`, след което съдържанието от буфера се парсва в двумерен масив `char (* students)[148]`, за който се заделя памет динамично. Функцията връща броя на студентите, прочетени от файла.
- **`void loadListData(node * head)`**: функцията приема като параметър главния възел на списъка. Целта на функцията е да състави списъка със студенти, като информацията за

студент от всеки елемент на `char (*students)[148]` се обработва и подава на новият елемент от свързания списък (**`void addNode(node * head, char * name, Gender gender)`**)

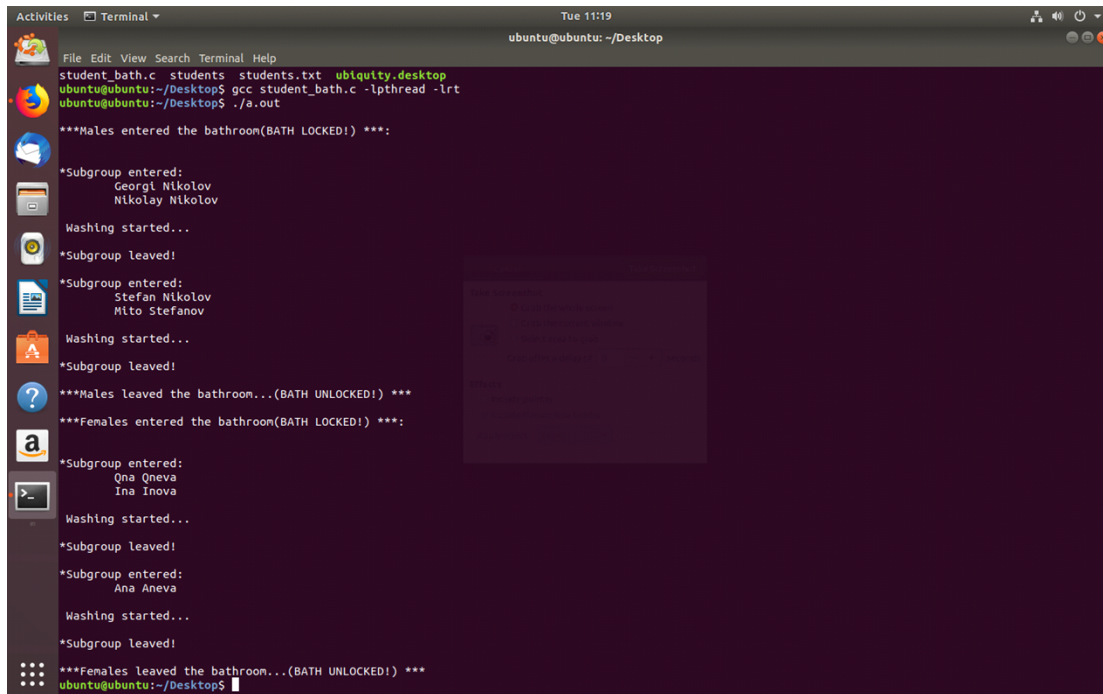
- **`void cleanString(char * str, char remove)`:**
спомогателна функция, трие даден символ от стринг.
- **`Gender validateStringToEnum(char * input)`:**
от подаден стринг връща изроим тип {MALE/FEMALE} в зависимост от стринга.
- **`bool isStudentWashed(char * name)` :**
функцията връща `true`, ако името на даден студент фигурира във файла с изкъпани студенти, `false` в противен случай.

Експериментални данни

1-ви опит: Файлът "students.txt" има следното съдържание:

```
Qna Qneva, FEMALE;  
Ina Inova, FEMALE;  
Ana Aneva, FEMALE;  
Georgi Nikolov, MALE;  
Nikolay Nikolov, MALE;  
Stefan Nikolov, MALE;  
Mito Stefanov, MALE;
```

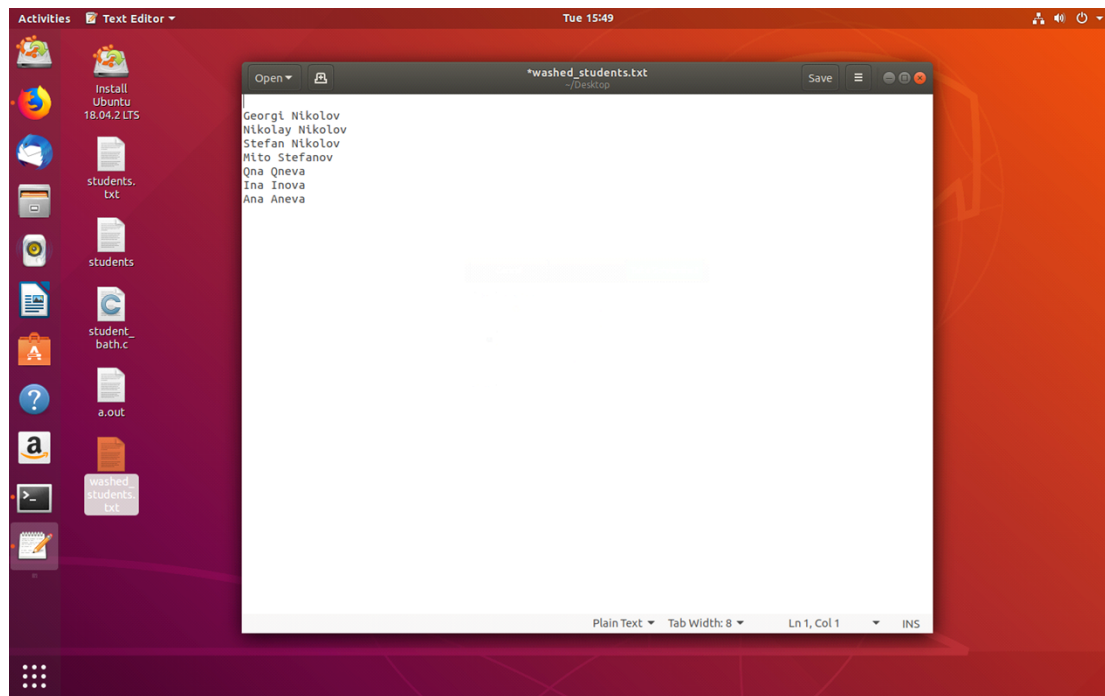
След пускане на програмата резултатът е:



```
student_bath.c  students  students.txt  ublquity.desktop  
ubuntu@ubuntu:~/Desktop$ gcc student_bath.c -lpthread -lrt  
ubuntu@ubuntu:~/Desktop$ ./a.out  
***Males entered the bathroom(BATH LOCKED!) ***  
*Subgroup entered:  
  Georgi Nikolov  
  Nikolay Nikolov  
Washing started...  
*Subgroup leaved!  
*Subgroup entered:  
  Stefan Nikolov  
  Mito Stefanov  
Washing started...  
*Subgroup leaved!  
***Males leaved the bathroom...(BATH UNLOCKED!) ***  
***Females entered the bathroom(BATH LOCKED!) ***:  
*Subgroup entered:  
  Qna Qneva  
  Ina Inova  
Washing started...  
*Subgroup leaved!  
*Subgroup entered:  
  Ana Aneva  
Washing started...  
*Subgroup leaved!  
***Females leaved the bathroom...(BATH UNLOCKED!) ***  
ubuntu@ubuntu:~/Desktop$
```

Както се вижда, първо са влезли мъжете, след което жените. Във файла: students_washed.txt е

записано следното

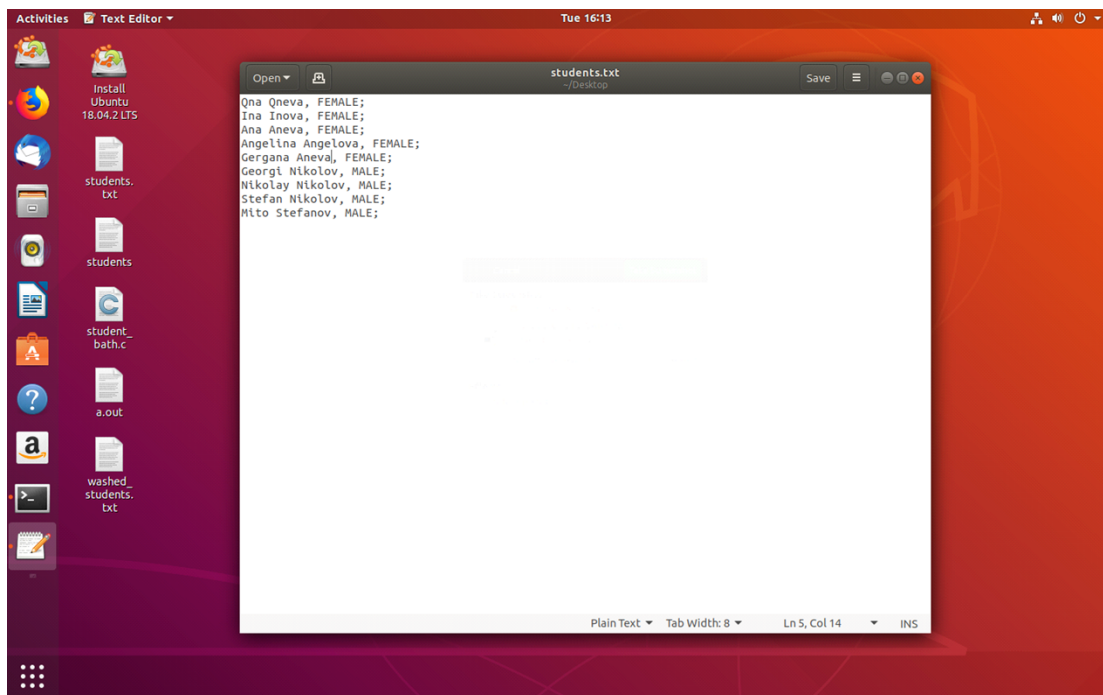


ВНИМАНИЕ!!! СЛЕД ПЪРВОНАЧАЛНОТО
ПУСКАНЕ НА ПРОГРАМАТА ИМА ОПАСНОСТ
ФАЙЛЪТ "washed_students.txt"

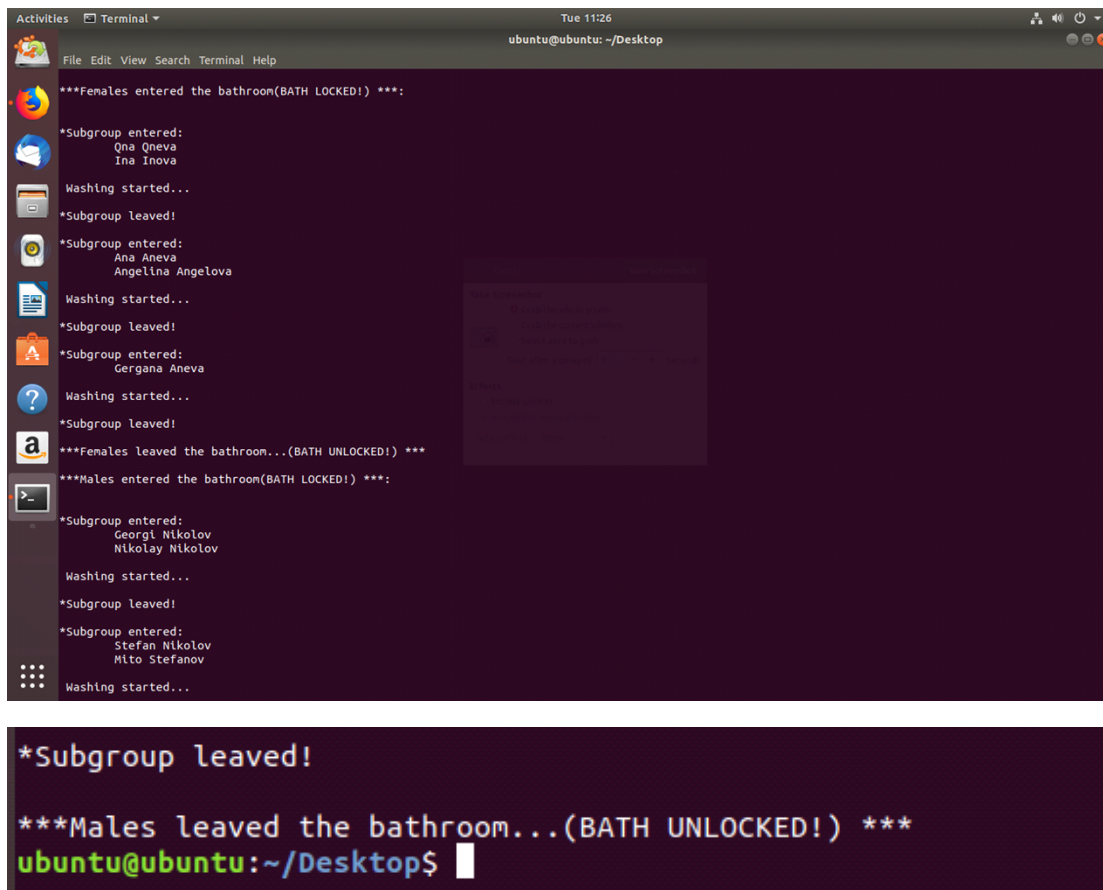
ДА БЪДЕ ЗАБРАНЕН ЗА ЧЕТЕНЕ И ЗАПИС. **ТОВА**
НАРУШАВА РАБОТАТА НА ПРОГРАМАТА!

ПРОБЛЕМЪТ СЕ ОПРАВЯ СЛЕД ДАВАНЕ НА
ПРАВА ОТ НАСТРОЙКИТЕ НА ФАЙЛА.

2-ри опит: Входният файл съдържа:



Резултатът от пускането на програмата ще бъде:



Източник(source code):

<https://github.com/GNNikolov/System-programming>